TP 1 Codeblock

MAIN :

```
#include <stdio.h>
#include <stdlib.h>
#include "biblio.h"

int main()
{
/*    int rayon,longeur,largeur;
   printf("rayon cercle : ");
   scanf("%d",&rayon);
   surface(rayon);
   printf("/n longeur et largeur rectangle : ");
   scanf("%d, %d",&longeur, &largeur);
   rectangle(longeur,largeur);*/

   matrice();
}
```

BIBLIO :

```
int li1,co1,li2,co2,li,co;

void surface(int r) {
   float s = 3.141592654*r*r;
   printf("La surface d'un cercle de rayon %d metres est de %f metres carres \n",r,s);
}

void rectangle(int lon, int lar) {
   int s = lon * lar;
   printf("La surface d'un rectangle de taille %d par %d metres est de %d metres carres \n",lon,lar,s);
```

```
}

void matrice() {
   printf("choisissez le nombre de lignes de la premiere matrice : ");
   scanf("%d",&li1);
   printf("choisissez le nombre de colonnes de la premiere matrice : ");
   scanf("%d",&co1);
   printf("choisissez le nombre de lignes de la deuxieme matrice : ");
   scanf("%d",&li2);
   printf("choisissez le nombre de colonnes de la deuxieme matrice : ");
   scanf("%d",&co2);
   int Tab1[li1][co1],Tab2[li2][co2];
   systeme_inverse(Tab1,Tab2);
}

void saisie (int li, int co, int t[li][co]) {
   for (int i = 0; i <= li-1; i++) {
      for (int j = 0; j <= co-1; j++) {
         printf("ligne %d, colonne %d : ",i+1,j+1);
         scanf("%d",&t[i][j]);
      }
   }
}

void affichage(int li, int co, int u[li][co]) {
   printf("Matrice :\n");
   for (int k = 0; k <= li-1; k++) {
      for (int l = 0; l <= co-1; l++) {
         printf("%d ",u[k][l]);
      }
      printf("\n");
```

```
    }
}

void affichagef(int li, int co, float u[li][co]) {
    printf("Matrice :\n");
    for (int k = 0; k <= li-1; k++) {
        for (int l = 0; l <= co-1; l++) {
            printf("%f ",u[k][l]);
        }
        printf("\n");
    }
}

void somme(int a[li1][co1], int b[li2][co2]) {
    if ((li1 == li2)&&(co1 == co2)) {
        int c[li1][co1];
        printf("Matrice 1 :\n");
        saisie(li1,co1,a);
        affichage(li1,co1,a);
        printf("Matrice 2 :\n");
        saisie(li2,co2,b);
        affichage(li2,co2,b);
        printf("Resultat :\n");
        for (int i = 0; i <= li1-1; i++) {
            for (int j = 0; j <= co1-1; j++) {
                c[i][j] = a[i][j] + b[i][j];
            }
        }
        affichage(li1,co1,c);
    }
    else printf("Les deux matrices doivent avoir la meme taille");
```

```
}

void difference(int a[li1][co1], int b[li2][co2]) {
   if ((li1 == li2)&&(co1 == co2)) {
      int c[li1][co1];
      printf("Matrice 1 :\n");
      saisie(li1,co1,a);
      affichage(li1,co1,a);
      printf("Matrice 2 :\n");
      saisie(li2,co2,b);
      affichage(li2,co2,b);
      printf("Resultat :\n");
      for (int i = 0; i <= li1-1; i++) {
         for (int j = 0; j <= co1-1; j++) {
            c[i][j] = a[i][j] - b[i][j];
         }
      }
      affichage(li1,co1,c);
   }
   else printf("Les deux matrices doivent avoir la meme taille");
}

void produit(int a[li1][co1], int b[li2][co2]) {
   if (co1 == li2) {
      int c[li1][co2];
      printf("Matrice 1 :\n");
      saisie(li1,co1,a);
      affichage(li1,co1,a);
      printf("Matrice 2 :\n");
      saisie(li2,co2,b);
      affichage(li2,co2,b);
```

```
        printf("Resultat :\n");
        for (int i = 0; i <= li2-1; i++) {
           for (int j = 0; j <= co1-1; j++) {
              c[i][j] = 0;
              for (int k = 0; k <= co1-1; k++) {
                 c[i][j] = c[i][j] + a[i][k] * b[k][j];
              }
           }
        }
        affichage(li1,co2,c);
    }
    else printf("La matrice 1 doit avoir autant de colonnes que de lignes de la matrice 2");
}

int determinant(int a[li1][co1]) {
    if ((li1==2)&&(co1==2)) {
        int det;
        printf("Matrice 1 :\n");
        saisie(li1,co1,a);
        affichage(li1,co1,a);
        det = (a[0][0]*a[1][1])-(a[0][1]*a[1][0]);
        printf ("Le determiannt vaut %d.",det);
        if (det == 0) printf (" La matrice n'est pas inversible.");
        else printf (" La matrice est inversible.");
    }
    else if ((li1==3)&&(co1==3)) {
        int det;
        printf("Matrice 1 :\n");
//      saisie(li1,co1,a);
//      affichage(li1,co1,a);
```

```
    det = a[0][0]*((a[1][1]*a[2][2])-(a[1][2]*a[2][1]))-a[1][0]*((a[0][1]*a[2][2])-(a[0][2]*a[2][1]))+a[2][0]*((a[0][1]*a[1][2])-(a[0][2]*a[1][1]));
    printf ("Le determiannt vaut %d.",det);
    if (det == 0) printf (" La matrice n'est pas inversible.");
    else printf (" La matrice est inversible.");
    return(det);
  }
  else printf("Pour le moment, le calcul du determinant n'est dispo qu'avec une matrice 2X2 ou 3X3.");
}

void inverse(int a[li1][co1]) {
  saisie(li1,co1,a);
  affichage(li1,co1,a);
  int deta;
  if ((li1==2)&&(co1==2)) {
    deta = determinant(a);
    if (deta==0) {
      printf("La matrice possede un determinant nul, elle ne peut etre inversee\n");
    }
    else {
      float c[2][2];
      c[0][0] = a[1][1];
      c[0][1] = -1*a[1][0];
      c[1][0] = -1*a[0][1];
      c[1][1] = a[0][0];
      printf("Resultat :\n");
      affichagef(2,2,c);
    }
  }
  else if ((li1==3)&&(co1==3)) {
    deta = determinant(a);
```

```
        if (deta==0) {
            printf("La matrice possede un determinant nul, elle ne peut etre inversee\n");
        }
        else {
            float coma[3][3], c[3][3];
            coma[0][0] = (a[1][1]*a[2][2])-(a[1][2]*a[2][1]);
            coma[0][1] = -1*((a[1][0]*a[2][2])-(a[1][2]*a[2][0]));
            coma[0][2] = (a[1][0]*a[2][1])-(a[1][1]*a[2][0]);
            coma[1][0] = -1*((a[0][1]*a[2][2])-(a[0][2]*a[2][1]));
            coma[1][1] = (a[0][0]*a[2][2])-(a[0][2]*a[2][0]);
            coma[1][2] = -1*((a[0][0]*a[2][1])-(a[0][1]*a[2][0]));
            coma[2][0] = (a[0][1]*a[1][2])-(a[0][2]*a[1][1]);
            coma[2][1] = -1*((a[0][0]*a[1][2])-(a[0][2]*a[1][0]));
            coma[2][2] = (a[0][0]*a[1][1])-(a[0][1]*a[1][0]);
            for(int i=0;i<=2;i++) {
                for(int j=0;j<=2;j++) {
                    c[i][j] = coma[j][i];
                    c[i][j] = c[i][j] * (1/(float)deta);
                }
            }
            printf("Resultat : 1/%d * \n",deta);
            affichagef(3,3,c);
        }
    }
    else printf("Pour le moment, le calcul de l'inverse n'est dispo qu'avec une matrice 2X2 ou 3X3.");

}

void systeme_cramer(int a[3][3], int b[3][1]) {
    saisie(li1,co1,a);
    affichage(li1,co1,a);
```

```
saisie(li2,co2,b);
affichage(li2,co2,b);
if ((co2==1)&&(li2==3)&&(li1==3)&&(co1==3)) {
   float deta, detx1, detx2, detx3, I1, I2, I3;
   int x1[3][3], x2[3][3], x3[3][3];
   for (int i = 0; i <= 2; i++) {
      for (int j = 0; j<= 2; j++)
      x1[i][j] = a[i][j];
   }
   for (int i = 0; i <= 2; i++) {
      x1[i][0] = b[i][0];
   }
   for (int i = 0; i <= 2; i++) {
      for (int j = 0; j<= 2; j++)
      x2[i][j] = a[i][j];
   }
   for (int i = 0; i <= 2; i++) {
      x2[i][1] = b[i][0];
   }
   for (int i = 0; i <= 2; i++) {
      for (int j = 0; j<= 2; j++)
      x3[i][j] = a[i][j];
   }
   for (int i = 0; i <= 2; i++) {
      x3[i][2] = b[i][0];
   }
   deta = determinant(a);
   detx1 = determinant(x1);
   detx2 = determinant(x2);
   detx3 = determinant(x3);
   I1 = detx1/deta;
```

```
        I2 = detx2/deta;
        I3 = detx3/deta;
        printf("Valeurs des inconnus : x = %f, y = %f, z = %f",I1,I2,I3);
    }
    else {
        printf("La matrice a doit avoir une taille de 3X3 et la matrice b une taille de 3X1\n");
        printf("Pour le moment, ne resous que les systemes de 3 equations a 3 inconnus.\n");
    }
}

void systeme_inverse(int a[3][3], int b[3][1]) {
    saisie(3,3,a);
    affichage(3,3,a);
    saisie(3,1,b);
    affichage(3,1,b);
    int deta;
    float coma[3][3], c[3][3], d[3][1];
    deta = determinant(a);
    if (deta==0) {
        printf("La matrice possede un determinant nul, elle ne peut etre inversee\n");
    }
    else {
        coma[0][0] = (a[1][1]*a[2][2])-(a[1][2]*a[2][1]);
        coma[0][1] = -1*((a[1][0]*a[2][2])-(a[1][2]*a[2][0]));
        coma[0][2] = (a[1][0]*a[2][1])-(a[1][1]*a[2][0]);
        coma[1][0] = -1*((a[0][1]*a[2][2])-(a[0][2]*a[2][1]));
        coma[1][1] = (a[0][0]*a[2][2])-(a[0][2]*a[2][0]);
        coma[1][2] = -1*((a[0][0]*a[2][1])-(a[0][1]*a[2][0]));
        coma[2][0] = (a[0][1]*a[1][2])-(a[0][2]*a[1][1]);
        coma[2][1] = -1*((a[0][0]*a[1][2])-(a[0][2]*a[1][0]));
        coma[2][2] = (a[0][0]*a[1][1])-(a[0][1]*a[1][0]);
```

```
        for(int i=0;i<=2;i++) {
            for(int j=0;j<=2;j++) {
                c[i][j] = coma[j][i];
                c[i][j] = c[i][j] * (1/(float)deta);
            }
        }
    }
    for (int i = 0; i <= li2-1; i++) {
            for (int j = 0; j <= co1-1; j++) {
                d[i][j] = 0;
                for (int k = 0; k <= co1-1; k++) {
                    d[i][j] = d[i][j] + c[i][k] * b[k][j];
                }
            }
        }
    printf("resultat :\n");
    affichagef(3,1,d);
}
```